UNIT- I

Cloud Computing Fundamentals: Definition of Cloud computing, Roots of Cloud Computing , Layers and Types of Clouds, Desired Features of a Cloud, Cloud Infrastructure Management, Infrastructure as a Service Providers, Platform as a Service Providers.

Computing Paradigms: High-Performance Computing, Parallel Computing, Distributed Computing, Cluster Computing, Grid Computing.

Introduction to Cloud Computing

“Cloud is a parallel and distributed computing system consisting of a collection ofinter-connected and virtualized computers that are dynamically provisioned and presented as one or more unified computing resources based on service- level agreements (SLA) established through negotiation between the service provider and consumers.”

“Clouds are a large pool of easily usable and accessible virtualized resources (suchas hardware, development platforms and/or services). These resources can be dynamically reconfigured to adjust to a variable load (scale), allowing also for an optimum resource utilization”

“This pool of resources is typically exploited by a pay-per-use model in which guarantees are offered by the Infrastructure Provider by means of customized Service Level Agreements.”

“Clouds are hardware based services offering compute, network, and storage capacity where Hardware management is highly abstracted from the buyer, buyers incur infrastructure costs as variable OPEX, and infrastructure capacityis highly elastic.”

Key characteristics of cloud computing

(1) the illusion of infinite computing resources;
(2) the elimination of an up-front commitment by cloud users;
(3) the ability to pay for use…as needed

The National Institute of Standards and Technology (NIST) characterizes **cloud computing** as “. . . a pay-per-use model for enabling available, convenient, on-demand network access to a shared pool of configurable computing resources (e.g. networks, servers, storage, applications, services) that can be rapidly provisioned and released with minimal management effort or service provider interaction”.

Most common characteristics which a cloud should have:
(i) pay-per-use (no ongoing commitment, utility prices); (ii) elastic capacity and the illusion of infinite resources; (iii) self-service interface; and (iv) resourcesthat is abstracted or virtualized.

Roots of Cloud Computing

The roots of clouds computing can be tracked by observing the advancement of several technologies, especially in hardware (virtualization, multi-core chips), Internet technologies (Web services, service-oriented architectures, Web 2.0),distributed computing (clusters, grids), and systems management (autonomic computing, data center automation).

Figure 1.1 shows the convergence of technology fields that significantly advanced and contributed to the advent of cloud computing.

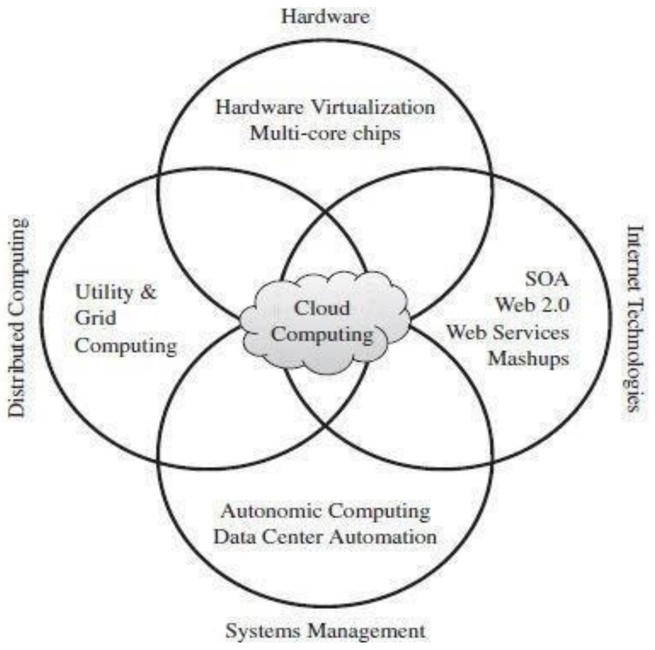


FIGURE 1.1. Convergence of various advances leading to the advent of cloud computing.

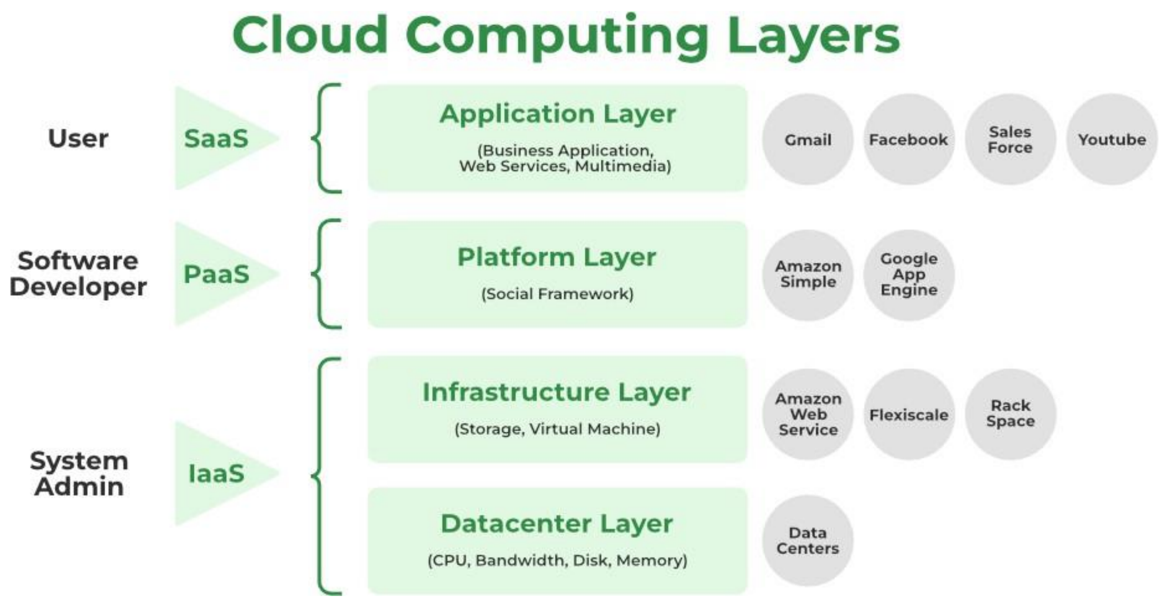
The IT world is currently experiencing a switch from in-house generated computing power into utility-supplied computing resources delivered over the Internet as Web services.

Computing delivered as a utility can be defined as “on demand delivery of infrastructure, applications, and business processes in a security-rich, shared, scalable, and based computer environment over the Internet for a fee”.

This model brings benefits to both consumers and providers of IT services. Consumers can attain reduction on IT-related costs by choosing to obtain cheaper services from external providers as opposed to heavily investing on IT infrastructure and personnel hiring. The “on-demand” component of this model allows consumers to adapt their IT usage to rapidly increasing or unpredictable computing needs.

Providers of IT services achieve better operational costs; hardware and software infrastructures are built to provide multiple solutions and serve many users, thus increasing efficiency and ultimately leading to faster return on investment (ROI) as well as lower total cost of ownership (TCO).

In the 1970s, companies who offered common data processing tasks, such as payroll automation, operated time-shared main frames as utilities, which could serve dozens of applications and often operated close to 100% of their capacity.
The mainframe era collapsed with the advent of fast and inexpensive microprocessors and IT data centers moved to collections of commodity servers. Apart from its clear advantages, this new model inevitably led to isolation of workload into dedicated servers, mainly due to incompatibilities between softwarestacks and operating systems.
In addition, the unavailability of efficient computer networks meant that IT infrastructure should be hosted in proximity to where it would be consumed. Altogether, these facts have prevented the utility computing reality of taking place on modern computer systems. These facts reveal the potential of delivering computing services with the speed and reliability that businesses enjoy with their local machines. The benefits of economies of scale and high utilization allow providers to offer computing services for a fraction of what it costs for a typical company that generates its own computing power.



SOA, Web Services, Web 2.0, and Mashups

The emergence of Web services (WS) open standards has significantly contributed to advances in the domain of software integration. Web services can combine together applications running on different messaging product platforms, enabling information from one application to be made available to others, and enabling internal applications to be made available over the Internet.

WS standards have been created on top of existing ubiquitous technologies such as HTTP and XML, thus providing a common mechanism for delivering services, making them ideal for implementing a service-oriented architecture (SOA). The purpose of a SOA is to address requirements of loosely coupled, standards-based, and protocol-independent distributed computing. In a SOA, software resources arepackaged as “services,” which are well-defined, self-contained modules that provide standard business functionality and are independent of the state or context of other services.

Services are described in a standard definition language and have a published interface. The maturity of WS has enabled the creation of powerful services that can be accessed on-demand, in a uniform way. An enterprise application that follows the SOA paradigm is a collection of services that together perform complex business logic.

In the consumer Web, information and services may be programmatically aggregated, acting as building blocks of complex compositions, called service mashups. Many service providers, such as Amazon, del.icio.us, Facebook, and Google, make their service APIs publicly accessible using standard protocols such as SOAP and REST. Consequently, one can put an idea of a fully functional Web application into practice just by gluing pieces with few lines of code.

In the Software as a Service (SaaS) domain, cloud applications can be built as compositions of other services from the same or different providers. Services such as user authentication, e-mail, payroll management, and calendars are examples of building blocks that can be reused and combined in a business solution in case a single, ready-made system does not provide all those features.

Grid Computing

Grid computing enables aggregation of distributed resources and transparently access to them. Most production grids such as Tera Grid and EGEE seek to share compute and storage resources distributed across different administrative domains, with their main focus being speeding up a broad range of scientific applications, such as climate modeling, drug design, and protein analysis.
A key aspect of the grid vision realization has been building standard Web services-based protocols that allow distributed resources to be "discovered, accessed, allocated, monitored, accounted for, and billed for, etc., and in general managed as a single virtual system." The Open Grid Services Architecture (OGSA) addresses this need for standardization by defining a set of core capabilities and behaviors that address key concerns in grid systems.

Globus Toolkit is a middleware that implements several standard Grid services andover the years has aided the deployment of several service-oriented Grid infrastructures and applications.

The development of standardized protocols for several grid computing activities has contributed—theoretically—to allow delivery of on-demand computing services over the Internet. However, ensuring QoS in grids has been perceived as a difficult endeavor. Lack of performance isolation has prevented grids adoption in a variety of scenarios, especially on environments where resources are oversubscribed or users are uncooperative.

Another issue that has lead to frustration when using grids is the availability of resources with diverse software configurations, including disparate operating systems, libraries, compilers, runtime environments, and so forth. At the same time, user applications would often run only on specially customized environments. Consequently, a portability barrier has often been present on most grid infrastructures, inhibiting users of adopting grids as utility computing environments

Utility Computing

In utility computing environments, users assign a “utility” value to their jobs, where utility is a fixed or time-varying valuation that captures various QoS constraints (deadline, importance, satisfaction). The valuation is the amount they are willing to pay a service provider to satisfy their demands. The service providersthen attempt to maximize their own utility, where said
utility may directly correlate with their profit. Providers can choose to prioritize high yield (i.e., profit per unit of resource) user jobs, leading to a scenario where shared systems are viewed as a marketplace, where users compete for resources based on the perceived utility or value of their jobs.

Hardware Virtualization

Cloud computing services are usually backed by large-scale data centers composed of thousands of computers. Such data centers are built to serve many users and host many disparate applications.
The idea of virtualizing a computer system's resources, including processors, memory, and I/O devices, has been well established for decades, aiming at improving sharing and utilization of computer systems. Hardware virtualization allows running multiple operating systems and software stacks on a single physicalplatform.

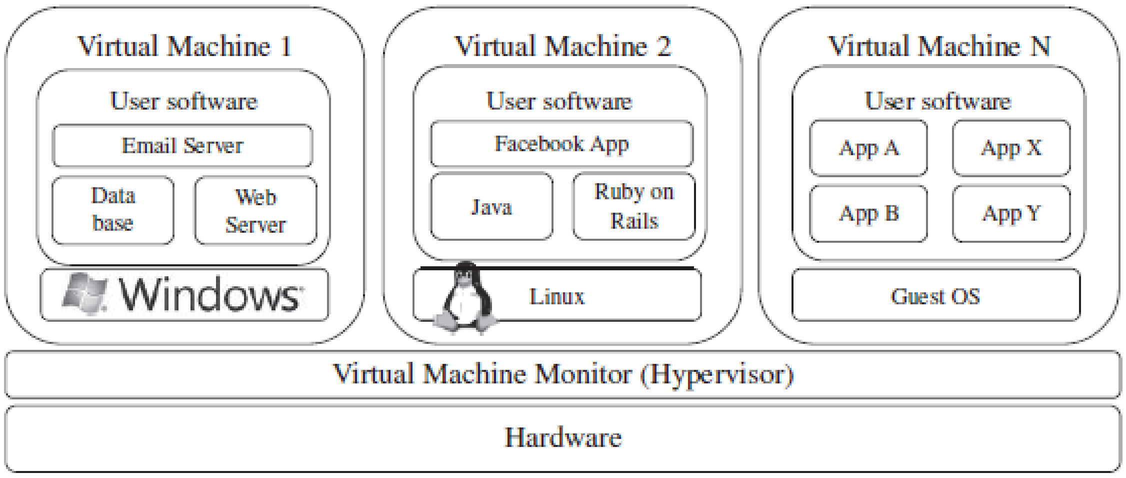


FIGURE 1.2. A hardware virtualized server hosting three virtual machines, each one running distinct operating system and user level software stack.

As depicted in Figure 1.2, a software layer, the virtual machine monitor (VMM), also called a hypervisor, mediates access to the physical hardware presenting to each guest operating system a virtual machine(VM), which is a set of virtual platform interfaces

The advent of several innovative technologies—multi-core chips, para- virtualization, hardware-assisted virtualization, and live migration of VMs—has contributed to an increasing adoption of virtualization on server systems.

Perceived benefits were improvements on sharing and utilization, better manageability, and higher reliability.

There are three basic capabilities regarding management of workload in a virtualized system, namely isolation, consolidation, and migration

Workload isolation is achieved since all program instructions are fully confined inside a VM, which leads to improvements in security. Better reliability is also achieved because software failures inside one VM do not affect others

The consolidation of several individual and heterogeneous workloads onto a single physical platform leads to better system utilization. This practice is also employed for overcoming potential software and hardware incompatibilities incase of upgrades, given that it is possible to run legacy and new operation systems concurrently

Workload migration, also referred to as application mobility, targets at facilitating hardware maintenance, load balancing, and disaster recovery. It is done by encapsulating a guest OS state within a VM and allowing it to be suspended, fully serialized, migrated to a different platform, and resumed immediately or preserved to be restored at a later date. A VM's state includes a full disk or partition image, configuration files, and an image of its RAM.

A number of VMM platforms exist that are the basis of many utility or cloud computing environments. The most notable ones are VMWare, Xen, andKVM.
VMWARE ESXi: is a VMM from VMWare. It is a bare-metal hypervisor,meaning that it installs directly on the physical server, whereas others may requirea host operating system. It provides advanced virtualization techniques ofprocessor, memory, and I/O. Especially, through page sharing, it can over commitmemory, thus increasing the density of VMs inside a single physical server.

Xen : The Xen hypervisor started as an open-source project and has served as a base to other virtualization products, both commercial and open-source. It has pioneered the para-virtualization concept, on which the guest operating system, by means of a specialized kernel, can interact with the hypervisor, thus significantly improving performance.

KVM: The kernel-based virtual machine (KVM) is a Linux virtualization subsystem. It has been part of the mainline Linux kernel since version 2.6.20, thus being natively supported by several distributions. In addition, activities such as memory management and scheduling are carried out by existing kernel features, thus making KVM simpler and smaller than hypervisors that take control of the entire machine

Virtual Appliances and the Open Virtualization Format

An application combined with the environment needed to run it (operating system, libraries, compilers, databases, application containers, and so forth) is referred toas a "virtual appliance."

Packaging application environments in the shape of virtual appliances eases software customization, configuration, and patching and improves portability. Most commonly, an appliance is shaped asa VM disk image associated with hardware requirements, and it can be readily deployed in a hypervisor. The VMWare virtual appliance marketplace allows users to deploy appliances onVMWare hypervisors or on partners public clouds, and Amazon allows developersto share specialized Amazon Machine Images (AMI) and monetize their usage on Amazon EC2.

In a multitude of hypervisors, where each one supports a different VM image format and the formats are incompatible with one another, a great deal of interoperability issues arises. In order to facilitate packing and distribution of software to be run on VMs several vendors, including VMware, IBM, Citrix, Cisco, Microsoft, Dell, and HP, have devised the Open Virtualization Format

(OVF). It aims at being “open, secure, portable, efficient and extensible” [32]. An OVF package consists of a file, or set of files, describing the VM hardware characteristics (e.g., memory, network cards, and disks), operating system details, startup, and shutdown actions, the virtual disks themselves, and other metadata containing product and licensing information. OVF also supports complex packages composed of multiple VMs.

Autonomic Computing

The increasing complexity of computing systems has motivated research on autonomic computing, which seeks to improve systems by decreasing human involvement in their operation. In other words, systems should manage themselves,with high-level guidance from humans

Autonomic, or self-managing, systems rely on monitoring probes and gauges (sensors), on an adaptation engine (autonomic manager) for computing optimizations based on monitoring data, and on effectors to carry out changes on the system. IBM’s Autonomic Computing Initiative has
contributed to define the four properties of autonomic systems: self-configuration, self-optimization, self-healing, and self-protection. IBM has also suggested a reference model for autonomic control loops of autonomic managers, called MAPE-K (Monitor Analyze Plan Execute—Knowledge)

The large data centers of cloud computing providers must be managed in anefficient way. In this sense, the concepts of autonomic computing inspiresoftware technologies for data center automation, which may perform taskssuch as: management of service levels of running applications; management ofdata center capacity; proactive disaster recovery; and automation of VMprovisioning

LAYERS AND TYPES OFCLOUDS

Cloud computing services are divided into three classes
(1) Infrastructure as a Service, (2) Platform as a Service, and (3) Softwareas aService

Figure 1.3 depicts the layered organization of the cloud stackfrom physical infrastructure to applications.

These abstraction levels can also be viewed as a layered architecture whereservices of a higher layer can be composed from services of the underlying layerA core middleware manages physical resources andthe VMs deployed on top of them; in addition, it provides the required features(e.g., accounting and billing) to offer multi-tenant pay- as-you-goservices.

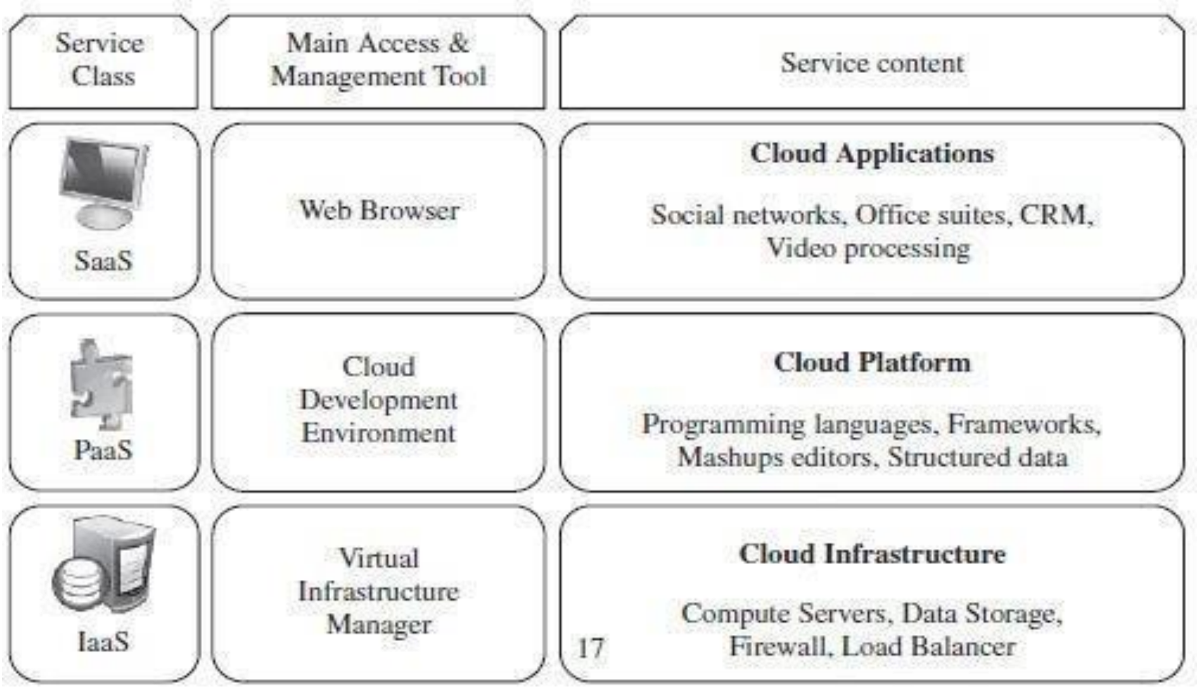

Service Class
Main Access & Management Tool
Service content
SaaS
Web Browser
Cloud Applications
Social networks, Office suites, CRM, Video processing
PaaS
Cloud Development Environment
Cloud Platform
Programming languages, Frameworks, Mashups editors, Structured data
IaaS
Virtual Infrastructure Manager
Cloud Infrastructure
Compute Servers, Data Storage, Firewall, Load Balancer
17

FIGURE 1.3 The cloud computing stack Infrastructure as a Service Offering virtualized resources (computation, storage, and communication) on demand is known as Infrastructure as a Service (IaaS). A cloud infrastructure enables on-demand provisioning of servers running several choices of operating systems and a customized software stack. Infrastructure services are considered to be the bottom layer of cloud computing systems.

Amazon Web Services mainly offers IaaS, which in the case of its EC2service means offering VMs with a software stack that can be customized similar to how an ordinary physical server would be customized. Users are given privileges to perform numerous activities to the server, such as: starting and stopping it, customizing it by installing software packages, attaching virtual disks to it, and configuring access permissions and firewalls rules.

Platform as a Service

A cloud platform offers an environment on which developers create and deploy applications and do not necessarily need to know how many processors or how much memory that applications will be using. In addition, multiple programming models and specialized services (e.g., data access, authentication, and payments) are offered as building blocks to new applications.

Google App Engine, an example of Platform as a Service, offers a scalable environment for developing and hosting Web applications, which should be written in specific programming languages such as Python or Java, and use the services' own proprietary structured object data store.

Software as a Service

Applications reside on the top of the cloud stack. Services provided by this layer can be accessed by end users through Web portals. Therefore, consumers are increasingly shifting from locally installed computer programs to on-line software services that offer the same functionally. Traditional desktop applications such as word processing and spreadsheet can now be accessed as a service in the Web. This model of delivering applications, known as Software as a Service (SaaS), alleviates the burden of software maintenance for customers and simplifies development and testing for providers.

Salesforce.com, which relies on the SaaS model, offers business productivity applications (CRM) that reside completely on their servers, allowing customers to customize and access applications on demand.

Deployment Models

A cloud can be classified as public, private, community, or hybrid based on model of deployment as shown in Figure 1.4.

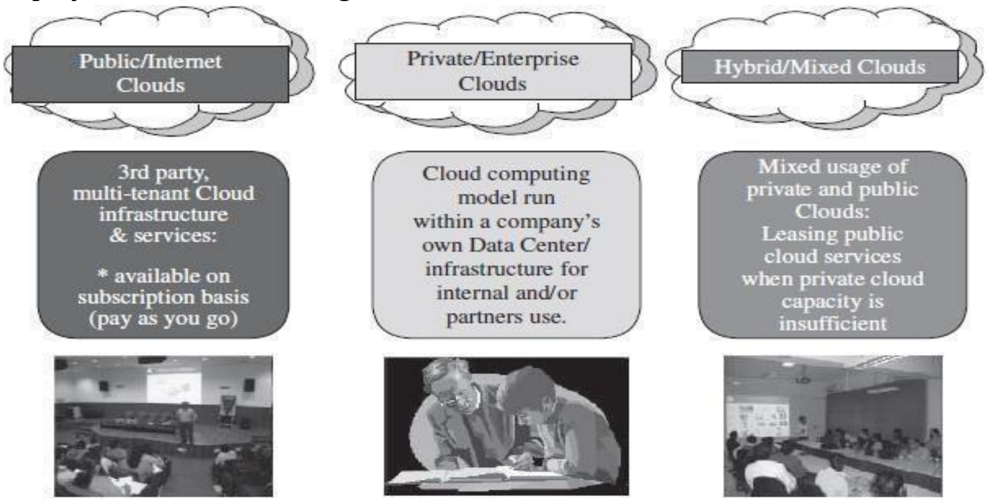

Public/Internet Clouds
Private/Enterprise Clouds
Hybrid/Mixed Clouds
3rd party, multi-tenant Cloud infrastructure & services:
* available on subscription basis (pay as you go)
Cloud computing model run within a company's own Data Center/ infrastructure for internal and/or partners use.
Mixed usage of private and public Clouds: Leasing public cloud services when private cloud capacity is insufficient

FIGURE 1.4. Types of clouds based on deployment models.

Public cloud: "cloud made available in a pay-as-you-go manner to the general public"
Private cloud: "internal data center of a business or other organization, not made available to the general public."
Community cloud:"shared by several organizations and supports a specific community that has shared concerns (e.g., mission, security requirements, policy, and compliance considerations) **Hybrid cloud** takes shape when a private cloud is supplemented with computing capacity from public clouds.
The approach of temporarily renting capacity to handle spikes in load is known as **"cloud- bursting"**